IBM Developer SKILLS NETWORK

Winning Space Race with Data Science

Olga Vovka
14 February 2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- Summary of methodologies
  1. Data collection through API
  2. Data collection with Web Scaping
  3. Data Wrangling
  4. Exploratory Data Analysis with SQL and Data Visualization
  5. Interactive Visual Analytics with Folium
  6. Machine Learning Prediction
- Summary of all results
- Data has been collected from public SpaceX API and by scrapping SpaceX Wikipedia page and used Beautiful soup library for it. For Falcon 9 used data from Forest Katsch, at zlsadesign.com
- Exploratory data analysis result. Interactive analytics in screenshots. Predictive Analytics result

Introduction

- Is space travel affordable for everyone?
- Space X advertises Falcon 9 rocket launches costs 62 million dollars and other providers cost upward of 165 million dollars each, much of the saving is because Space X can reuse the first stage. This information can be used if an company wants to bid against Space X for a rocket launch. In this capstone, we will predict if the Falcon 9 the first stage will land successfully by using machine learning pipeline.
- After determining the cost of launch and SpaceX the first stage reuse, we can predict if Spaces X's Falcon 9 launch could be regular rockets.
- We should find answer for the question: factors affecting the rocket successfully lending.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Collecting the Data with the Application Programming Interface (API): Space X API and scraping Wikipedia
  - Perform data wrangling
  - Wrangling Data using an API, Sampling Data, and Dealing with Nulls.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Classification models building, tuning, evaluation.

Data Collection

- Data was collected by using get request to the SpaceX API
- Then we decoded the response content using json function and turn it into pandas dataframe
- Then data was cleanded and fill in missing values
- Performed web scraping from Wikipedia for Falcon 9 launch record with BeautifulSoup
- The objective was to extract the launch records as html table, parse the table and convert it to a pandas dataframe for future analysis

Data Collection – SpaceX API

- SpaceX REST API:

1. Past data in request form

https://api.spacexdata.com/v4/launches/past

2. Booster data is gained from a rocket type request

https://api.spacexdata.com/v4/rockets/

3. Lanchpad name, longityde and latitude comes from a launch site request

https://api.spacexdata.com/v4/launchpads/

4. Payload data comes from a payloads request

https://api.spacexdata.com/v4/payloads/

5. Specific rocket core information comes from a core request

https://api.spacexdata.com/v4/core/

- Falcon 9 data was added to a pandas dataframe, it was exported to a csv file.
https://github.com/olgavovka/testrepo/blob/main/SpaceX_csv.csv

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

1. Used the requested library to scape data from

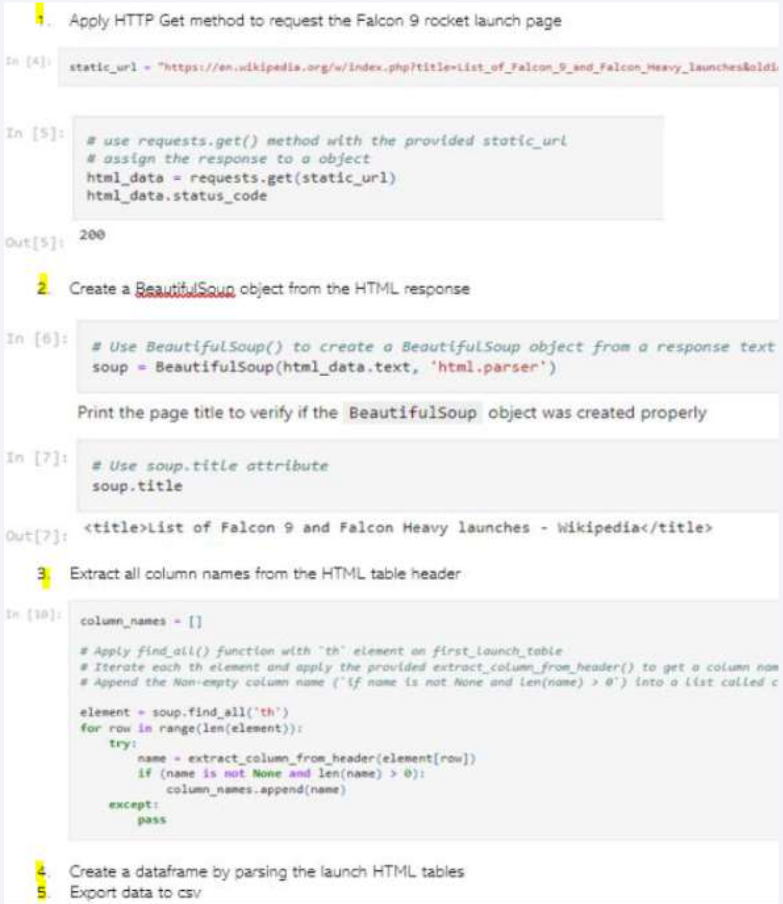
https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

2. Used BeautifulSoup to parse the content returned in the response

3. The parsed data was added to a pandas dataframe and then exported to a cvs file

https://github.com/olgavovka/testrepo/blob/main/SpaceX.ipynd

https://labs.cognitiveclass.ai/v2/tools/jupyterlab?ulid=ulid-d1931387220a35499622e3ac1c2f516f0e5c99d1

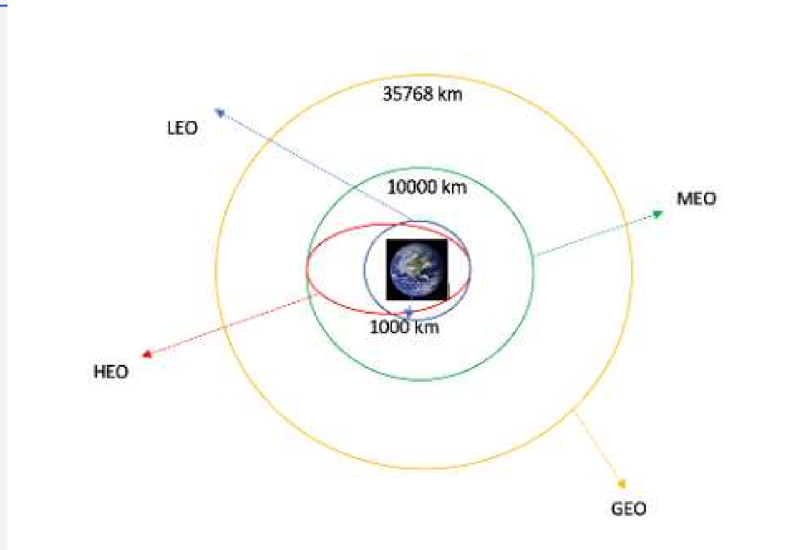


Data Wrangling

Outcomes convert to Classes y. y. (either 0 or 1). 0 is a bad outcome, that is, the booster did not land. 1 is a good outcome, that is, the booster did land. The variable Y will represent the classification variable that represents the outcome of each launch.

A classification variable class was created to encode the landing outcome as either 0 bad or 1 good. The variable is the target variable that the classification algorithm will need to predict.

https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/08d40490-8359-4c47-ae36-57e39f156b9f?projectid=55fc085c-dac5-448f-a36f-1285d45b8a07&context=cpdaas

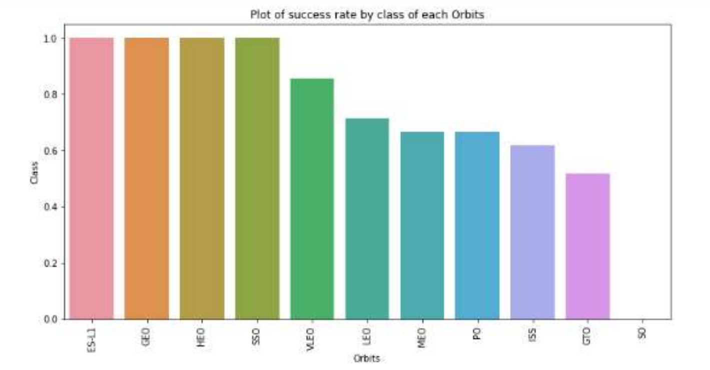



```
[15]: df["Class"].mean()

[15]: 0.6666666666666666
```

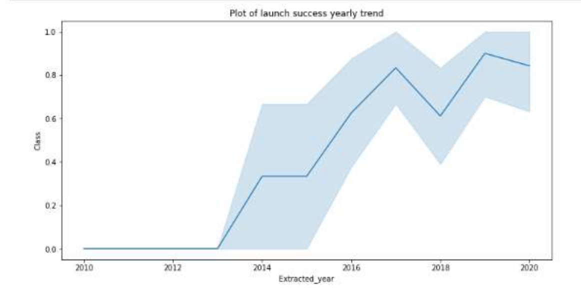
EDA with Data Visualization

- Perform exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib.
- The relationship between Flight number, Launch site, Payload mass, orbit type were investigating using a variety of graphs type, such as bar graphs, scatter plots, line graphs.
- All visualizations were color-code by class and effect of variables on the launch outcome were visible.
- All data was cast to a float6 type.
- https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/0f0a1050-a2a2-4296-904a-6fcd0761ad63?projectid=55fc085c-dac5-448f-a36f-1285d45b8a07&context=cpdaas

```
# Use groupby method on Orbit column and get the mean of Class column
grouped_orbits = df.groupby(by=['Orbit'])['Class'].mean().sort_values(ascending=False).reset_index()
fig, ax=plt.subplots(figsize=(12,6))
ax = sns.barplot(x = 'Orbit', y = 'Class', data=grouped_orbits)
ax.set_title('Plot of success rate by class of each Orbits', fontdict={'size':12})
ax.set_ylabel('Class', fontsize = 10)
ax.set_xlabel('Orbits', fontsize = 10)
ax.set_xticklabels(ax.get_xticklabels(), fontsize = 10, rotation=90);
```




```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df_copy = df.copy()
df_copy['Extracted_year'] = pd.DatetimeIndex(df['Date']).year

# plot line chart
fig, ax=plt.subplots(figsize=(12,6))
sns.lineplot(data=df_copy, x='Extracted_year', y='Class')
plt.title('Plot of launch success yearly trend');
plt.show()
```



EDA with SQL

The following scripts were performed:

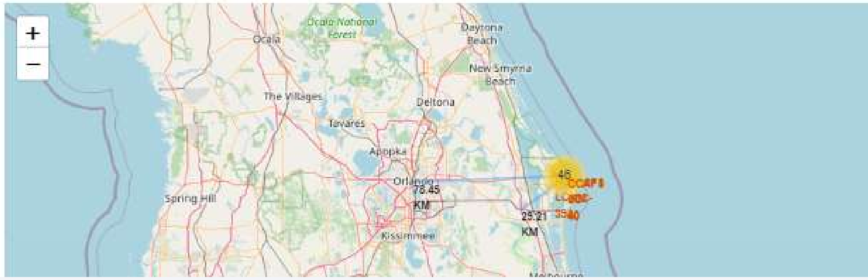
1. select distinct Launch_Site from SPACEXTBL;
2. select * from SPACEXTBL where Launch_Site like 'CCA%';
3. select SUM(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)';
4. select AVG(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1';
5. select MIN(date) as Firs_Successfull_landing_date from SPACEXTBL where LANDING__OUTCOME = 'Success (ground pad)';
6. select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ between 4000 and 6000 and LANDING__OUTCOME = 'Success (drone ship)';
7. select MISSION_OUTCOME, count(*) from SPACEXTBL group by MISSION_OUTCOME;
8. select distinct BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL) ;
9. select substr(Date,7,4) as YEAR, substr(Date,4,2) as MONTH, LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL where substr(Date,7,4)= '2015' and LANDING__OUTCOME = 'Success (drone ship)';
10. select LANDING__OUTCOME, count(*) from SPACEXTBL Where date between '2010-06-04' and '2017-03-20' group by LANDING__OUTCOME order by (Count(*)) DESC;

Build an Interactive Map with Folium

- Circle and Markers were added to aFolium map to indicate launch locations.
- MarketCluster was added for each site to visualize the launch outcomes at each site
- A line was drawn between a launch site and the coast. Label with the distance was added to show the proximity of the two.
- https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/61128471-cb50-4f56-bac2-5a1b52692b02?projectid=55fc085c-dac5-448f-a36f-1285d45b8a07&context=cpdaas

In [21]:
```
#Distance to Florida City

coordinates = [
    [28.56342, -80.57674],
    [28.5383, -81.3792]]

lines=folium.PolyLine(locations=coordinates, weight=1)
site_map.add_child(lines)
distance = calculate_distance(coordinates[0][0], coordinates[0][1], coordinates[1][0], coordinates[1][1])
distance_circle = folium.Marker(
    [28.5383, -81.3792],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#252526;"><b>%s</b></div>' % "{:10.2f} KM".format(distance),
        )
    )
site_map.add_child(distance_circle)
site_map
```

Out[21]:



Build a Dashboard with Plotly Dash

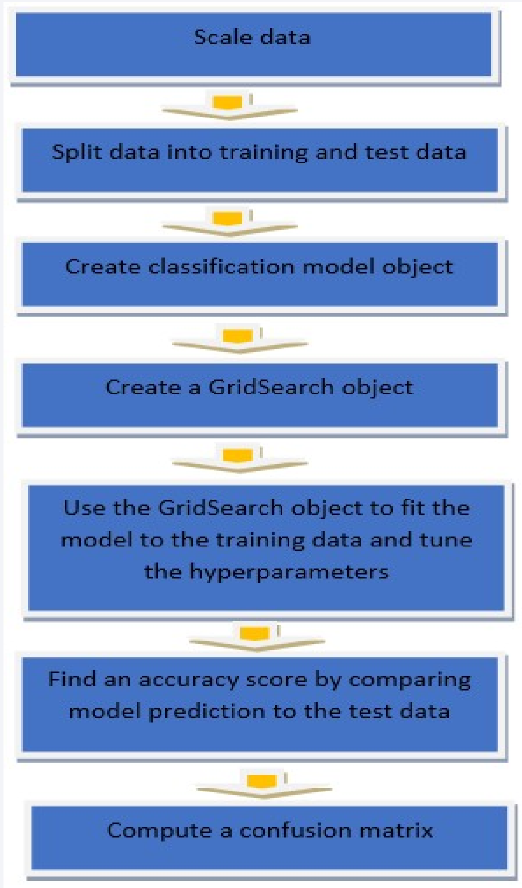
- This dashboard application contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart.
- Interactive dashboard was created to allow users to investigate the effects of launch site, payload mass and booster type on the launch outcome.
- Pie chart showed the successful outcome for all launch sites or the proportion of good and bad outcomes for any one selected launch site.
- Scatter chart showed how the launch outcome varied by the selected site and payload range and the data points were color-coded by booster type.

https://github.com/olgavovka/testrepo/blob/main/Plotly_Dash.txt

Create a NumPy array from the column Class in data

Predictive Analysis (Classification)

- The process in the flowchart for following classification algorithms:
- Logistic regression
- Support vector machines
- Decision tree
- K-nearest neighbors
- https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/490d6942-efcd-4cf1-a01c-f8f57cdd8697?projectid=55fc085c-dac5-448f-a36f-1285d45b8a07&context=cpdaas



Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

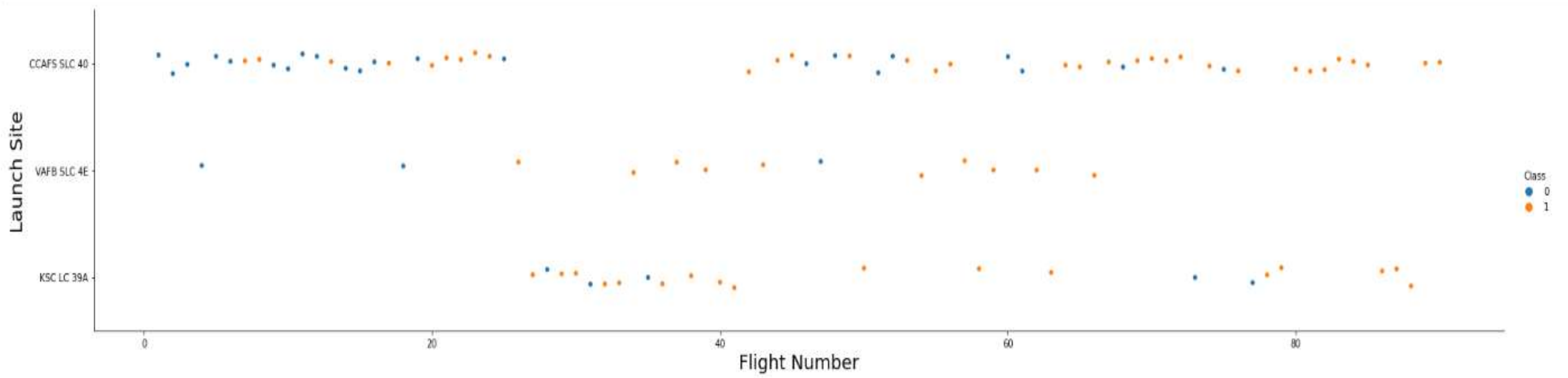
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

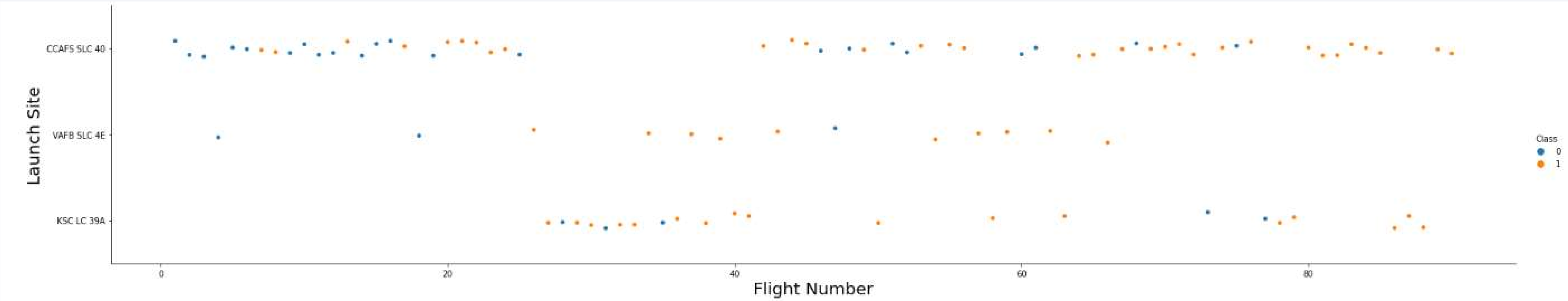
- Using the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```



Payload vs. Launch Site

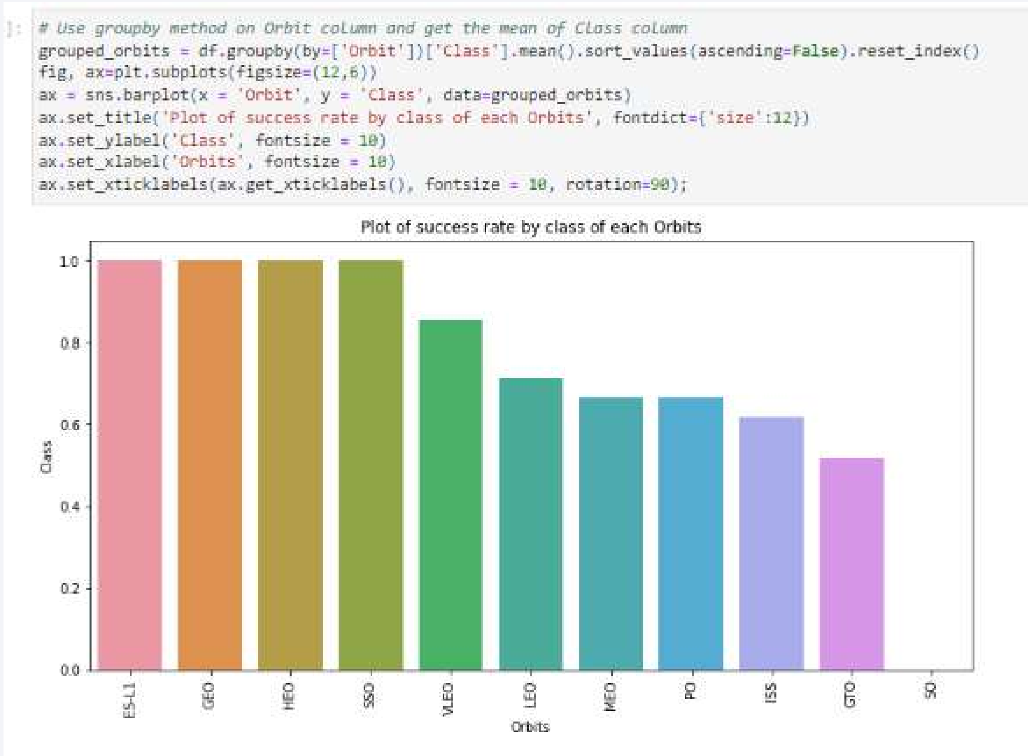
- The greatest the playload mass for site CCAFS SLC 40 the higher the success rate for the rocket



Success Rate vs. Orbit Type

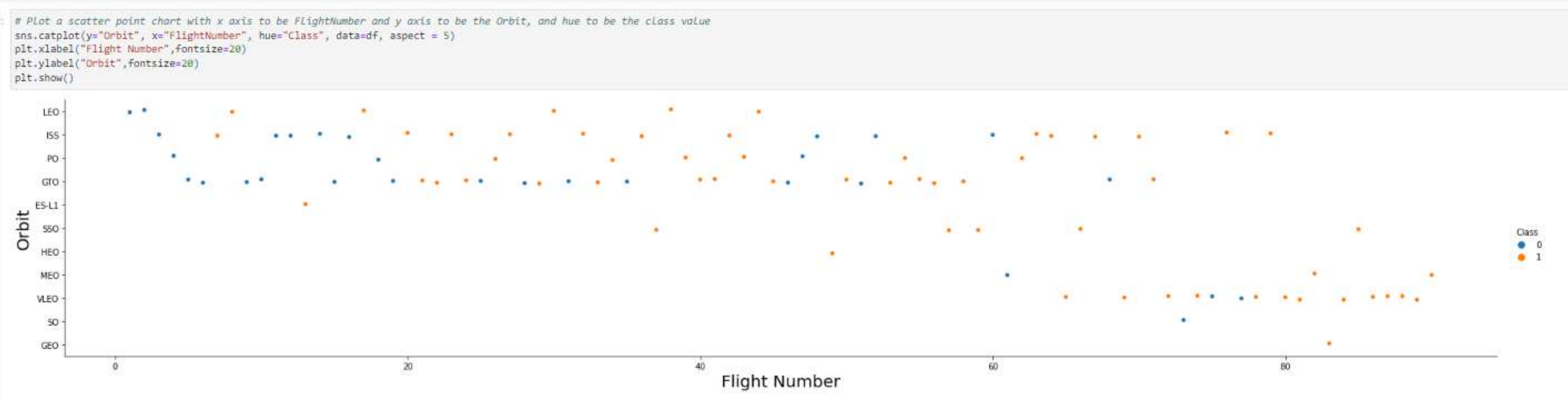
The most success rate are

1. ES-L-1
2. GEO
3. HEO
4. SSO
5. VLEO



Flight Number vs. Orbit Type

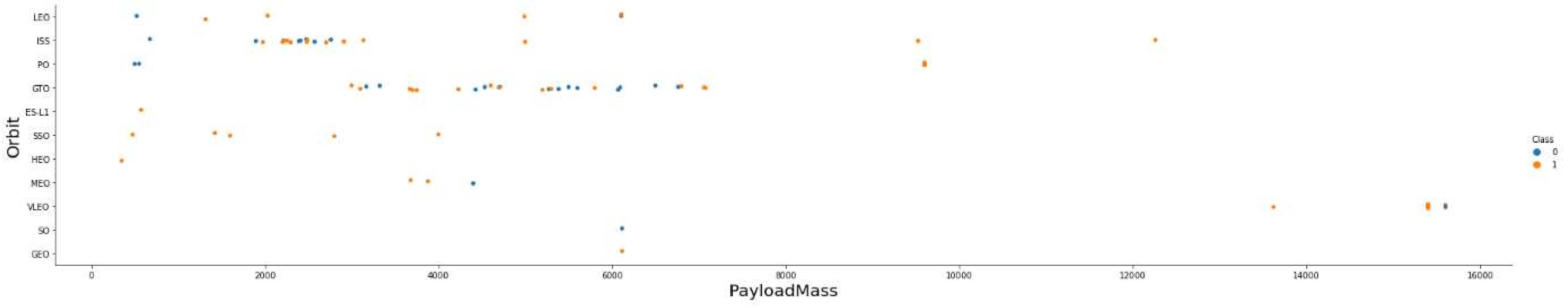
- From this plot we can suggest that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



Payload vs. Orbit Type

- The successful landing are from PO, LEO and ISS orbit.

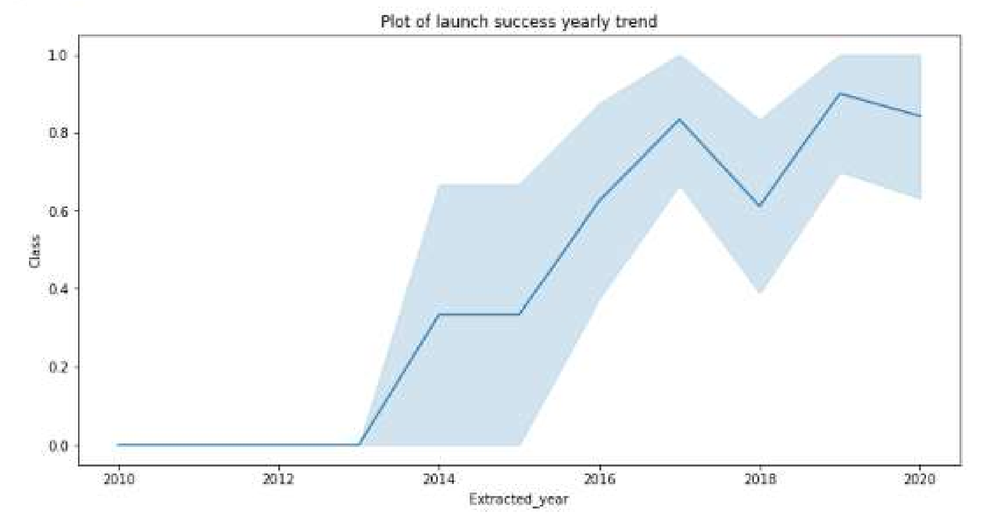
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

- We can see, that success rate since 2013 kept on increasing till 2020,

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df_copy = df.copy()
df_copy['Extracted_year'] = pd.DatetimeIndex(df['Date']).year

# plot line chart
fig, ax=plt.subplots(figsize=(12,6))
sns.lineplot(data=df_copy, x='Extracted_year', y='Class')
plt.title('Plot of launch success yearly trend');
plt.show()
```



All Launch Site Names

- Using following script we got Launch site names

select distinct Launch_Site from SPACEXTBL;

LAUNCH_SITE
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- We used the following script:

*select * from SPACEXTBL where Launch_Site like 'CCA%' limit 5;*

DATE	TIME__UTC_	BOOSTER_VERSION	LAUNCH_SITE	PAYLOAD	PAYLOAD_MASS__KG_	ORBIT	CUSTOMER	MISSION_OUTCOME	LANDING__OUTCOME
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-12	22:41:00	F9 v1.1	CCAFS LC-40	SES-8	3170	GTO	SES	Success	No attempt

Total Payload Mass

- For calculation the total payload carried by boosters from NASA we use:

select SUM(PAYLOAD_MASS__KG_) from SPACEXTBL

where CUSTOMER = 'NASA (CRS)';

1
22007

Average Payload Mass by F9 v1.1

- For calculate the average payload mass carried by booster version F9 v1.1 we used:

select AVG(PAYLOAD_MASS__KG_) from SPACEXTBL

where BOOSTER_VERSION = 'F9 v1.1';

1
3676

First Successful Ground Landing Date

For finding the dates of the first successful landing outcome on ground pad we used:

```
select MIN(date) as Firs_Successfull_landing_date
from SPACEXTBL
where LANDING__OUTCOME = 'Success (ground pad)';
```

FIRS_SUCCESSFULL_LANDING_DATE
2017-01-05

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

select BOOSTER_VERSION from SPACEXTBL

where PAYLOAD_MASS__KG_ between 4000 and 6000 and LANDING__OUTCOME = 'Success (drone ship)';

BOOSTER_VERSION
F9 FT B1022
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

For calculation the total number of successful and failure mission outcomes we used:

select MISSION_OUTCOME, count(*) from SPACEXTBL

group by MISSION_OUTCOME;

MISSION_OUTCOME	2
Success	44
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass:

```
select distinct BOOSTER_VERSION from SPACEXTBL
where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_)
from SPACEXTBL) ;
```

BOOSTER_VERSION
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1058.3
F9 B5 B1060.2

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

select substr(Date,7,4) as YEAR, substr(Date,4,2) as MONTH, LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE

from SPACEXTBL where substr(Date,7,4)= '2015'

and LANDING__OUTCOME = 'Success (drone ship)';

BOOSTER_VERSION	LANDING__OUTCOME	LAUNCH_SITE
F9 v1.1 B1012	Failure (drone ship)	CCAFS LC-40

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
select LANDING__OUTCOME, count(*) from SPACEXTBL
Where date between '2010-06-04' and  '2017-03-20'
 group by LANDING__OUTCOME order by (Count(*)) DESC;
```

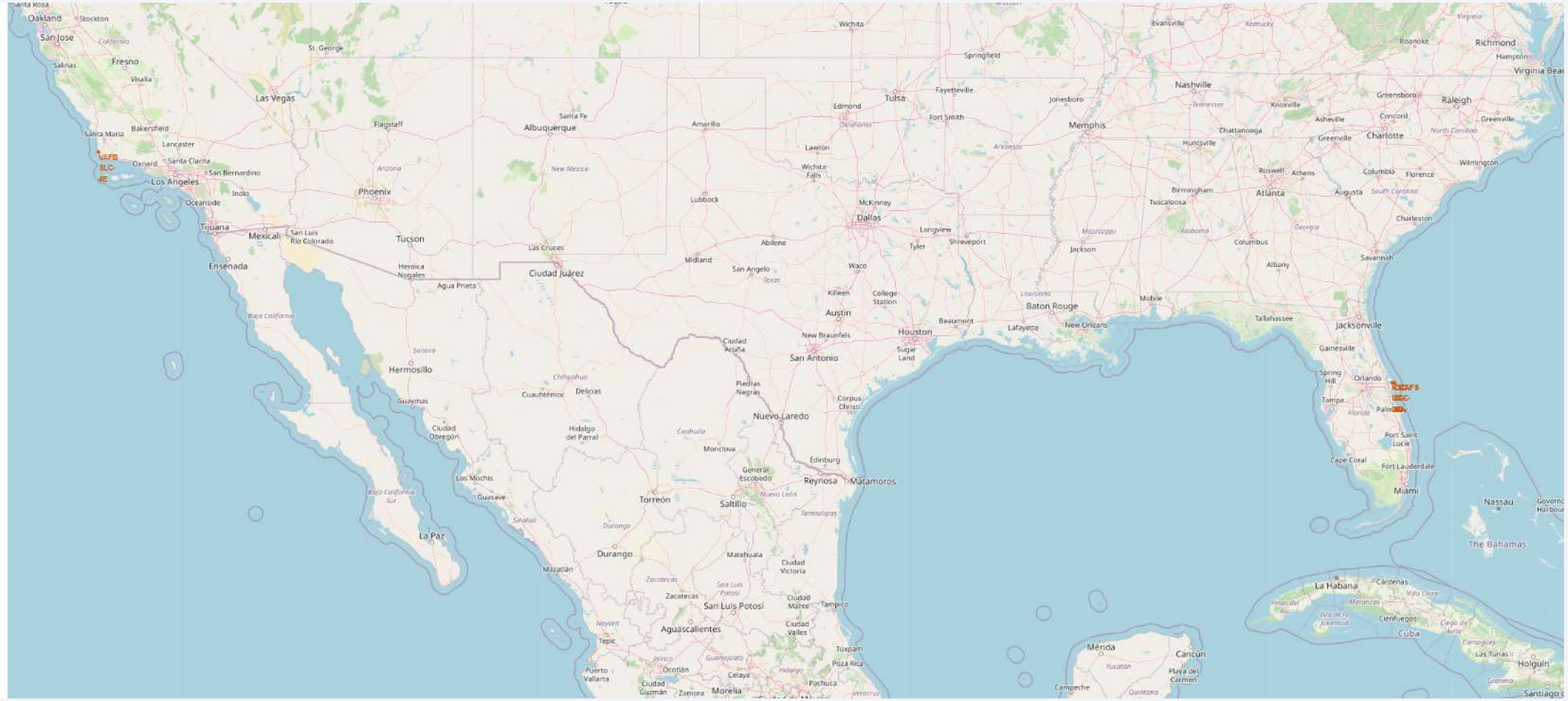
LANDING__OUTCOME	2
No attempt	7
Failure (drone ship)	2
Success (drone ship)	2
Success (ground pad)	2
Controlled (ocean)	1
Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

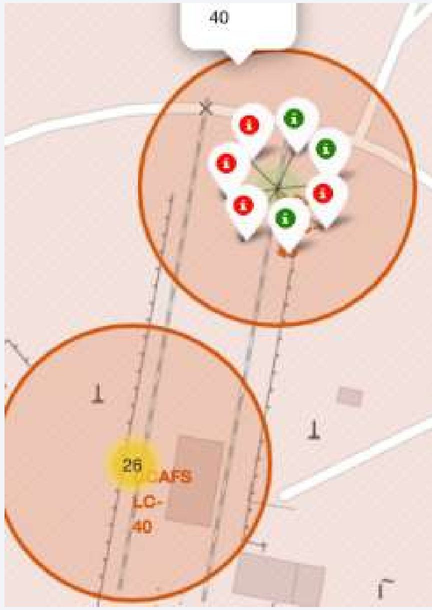
Global Map Markers

- We can see that the SpaceX launch sites are in the United States America coasts. Florida and California.



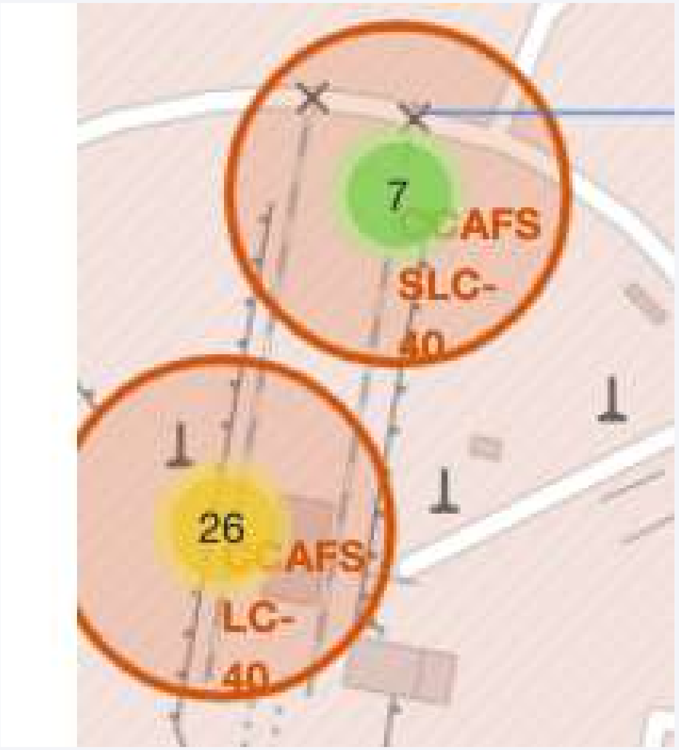
Markers showing launch sites with color labels

- Green Marker shows successful launches and Red Marker shows failures



Launch Site distance to landmarks

- Distance to coast. We an conclude, that it is closed to coast.

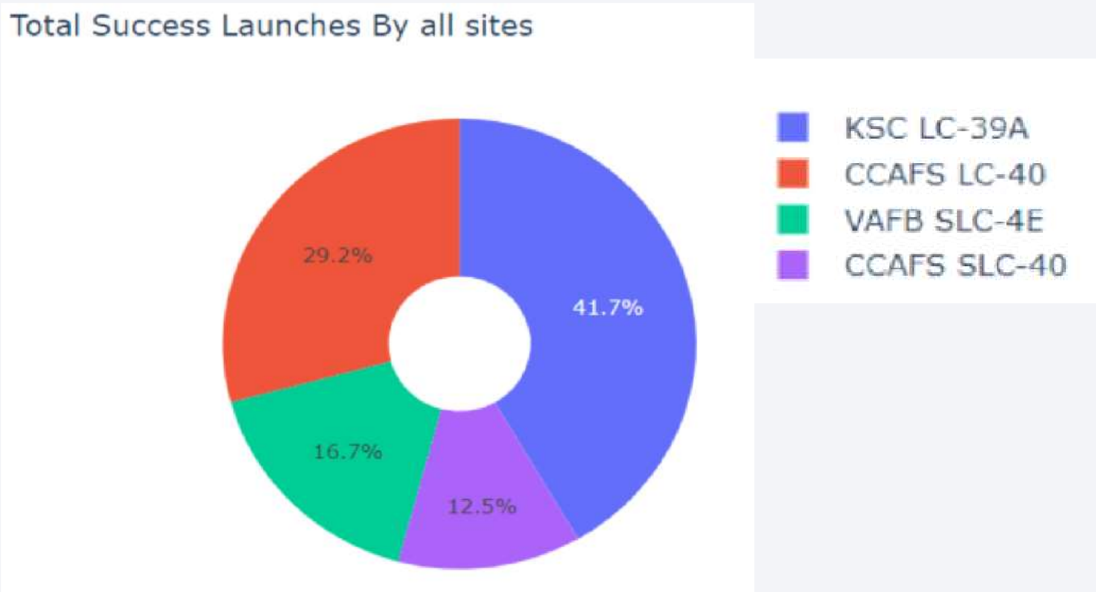


Section 4

Build a Dashboard with Plotly Dash

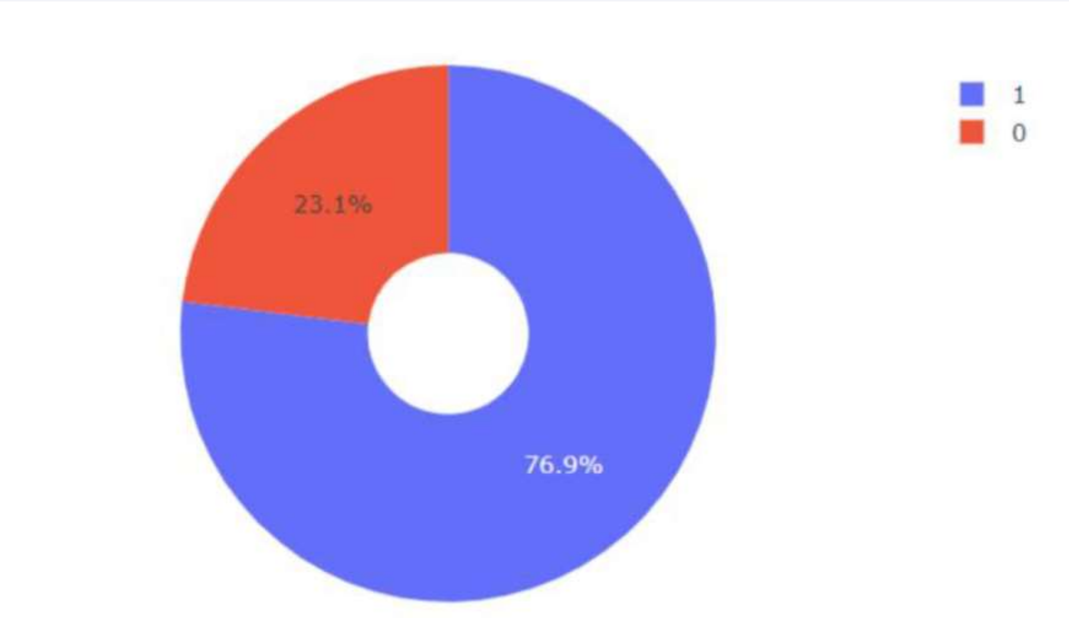
Pie chart showing the success percentage achieved by each launch site

- KSC LC-39A had the most successful launches from all the sites



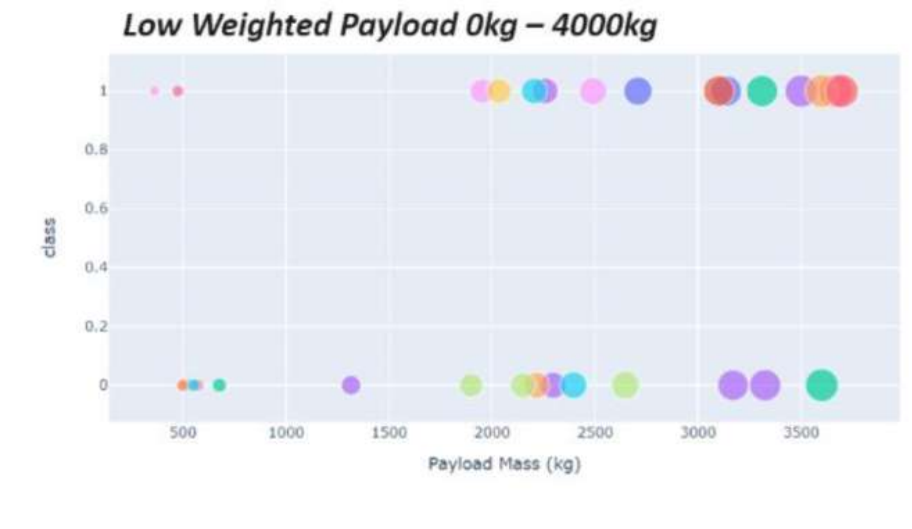
Pies chart showing the launch site with the highest launch success ratio

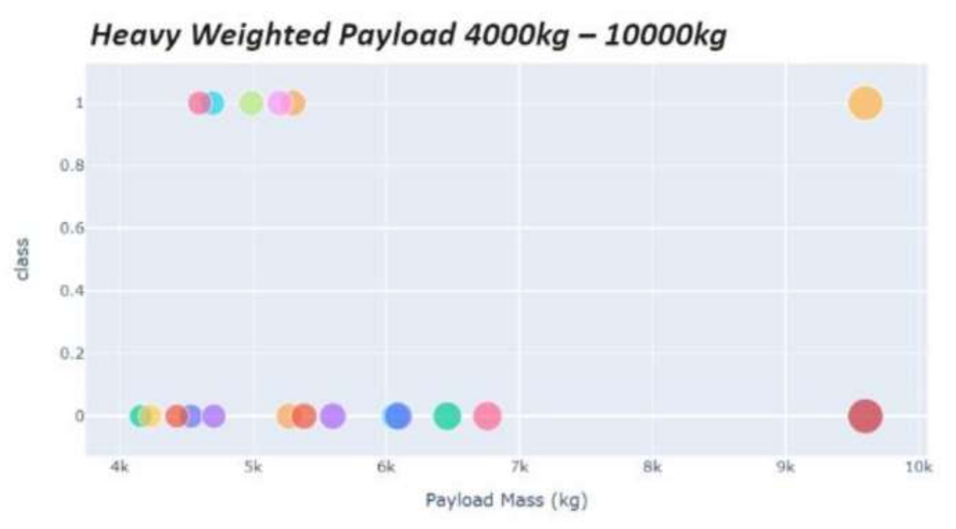
- KSL LC-39A achieved a 76,9% success rate while getting a 23,1% failure rate.



Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

- The success rates for low weighted payloads is higher than the heavy weighted payloads





Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The decision tree classifier is the model with the highest classification accuracy.

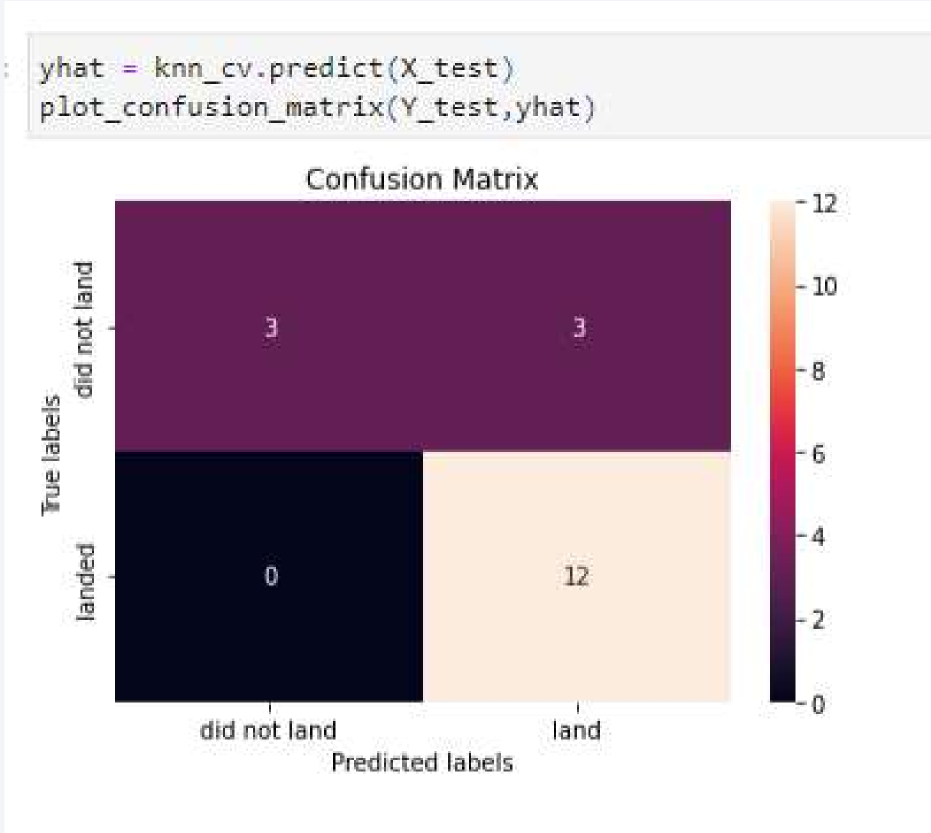
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'spl
itter': 'random'}
```

Confusion Matrix

The confusion matrix of the decision tree classifier shows that the classifier can distinguish between the different classes.

The main problem is the false positives, unsuccessful landing marked as successful landing by the classifier. best performing model with an explanation



Conclusions

- The larger the flight amount at launch site, the greater the success rate at a launch site.
- Orbit ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for risk task.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project


Thank you!